Algoritmos de Inteligência Artificial para Clustericação [24E4_2]

PROJETO DA DISCIPLINA - ENTREGA FINAL

Aluno: Marcio Feldmann

Repositório Público GitHub:
https://github.com/marciofeld/infnet_algoritmos_clusterizacao

Parte 1 - Infraestrutura

Para as questões a seguir, você deverá executar códigos em um notebook Jupyter, rodando em ambiente local, certifique-se que:

1. Você está rodando em Python 3.9+
2. Você está usando um ambiente virtual: Virtualenv ou Anaconda
3. Todas as bibliotecas usadas nesse exercícios estão instaladas em um ambiente virtual específico
4. Gere um arquivo de requerimentos (requirements.txt) com os pacotes necessários. É necessário se certificar que a versão do pacote está disponibilizada.
5. Tire um printscreen do ambiente que será usado rodando em sua máquina.
6. Disponibilize os códigos gerados, assim como os artefatos acessórios (requirements.txt) e instruções em um repositório GIT público. (se isso não for feito, o diretório com esses arquivos deverá ser enviado compactado no moodle).

In [1]:
```
# Parte 1 – Importação de bibliotecas

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.cluster import KMeans
from sklearn.cluster import AgglomerativeClustering
from scipy.spatial.distance import cdist
from scipy.cluster.hierarchy import dendrogram, linkage
```

Parte 2 - Escolha de base de dados

Para as questões a seguir, usaremos uma base de dados e faremos a análise exploratória dos dados, antes da clusterização.

1. Baixe os dados disponibilizados na plataforma Kaggle sobre dados sócio-econômicos e de saúde que determinam o índice de desenvolvimento de um país. Esses dados estão disponibilizados através do link: https://www.kaggle.com/datasets/rohan0301/unsupervised-learning-on-country-data
2. Quantos países existem no dataset?
3. Mostre através de gráficos a faixa dinâmica das variáveis que serão usadas nas tarefas de clusterização. Analise os resultados mostrados. O que deve ser feito com os dados antes da etapa de clusterização?
4. Realize o pré-processamento adequado dos dados.

In [2]:
```
# Parte 2 - Importação da Base de Dados
# INPUT: Country-data.csv

dataset = pd.read_csv('Country-data.csv')
dataset.head()
```

Out[2]:

	country	child_mort	exports	health	imports	income	inflation	life_expec
0	Afghanistan	90.2	10.0	7.58	44.9	1610	9.44	56.2
1	Albania	16.6	28.0	6.55	48.6	9930	4.49	76.3
2	Algeria	27.3	38.4	4.17	31.4	12900	16.10	76.5
3	Angola	119.0	62.3	2.85	42.9	5900	22.40	60.1
4	Antigua and Barbuda	10.3	45.5	6.03	58.9	19100	1.44	76.8

In [3]:
```
# Parte 2 - Exercício 2

qnt_paises = dataset['country'].nunique()
print('\033[1mParte 2 - Exercício 2 - Quantos países existem no dataset?\
print(f'Resposta do Exercício 2: \033[1m{qnt_paises}\033[0m')
```

Parte 2 - Exercício 2 - Quantos países existem no dataset?

Resposta do Exercício 2: **167**

In [4]:
```
# Definição das colunas de dados

colunas_de_dados = ['child_mort','exports','health','imports','income','i
```

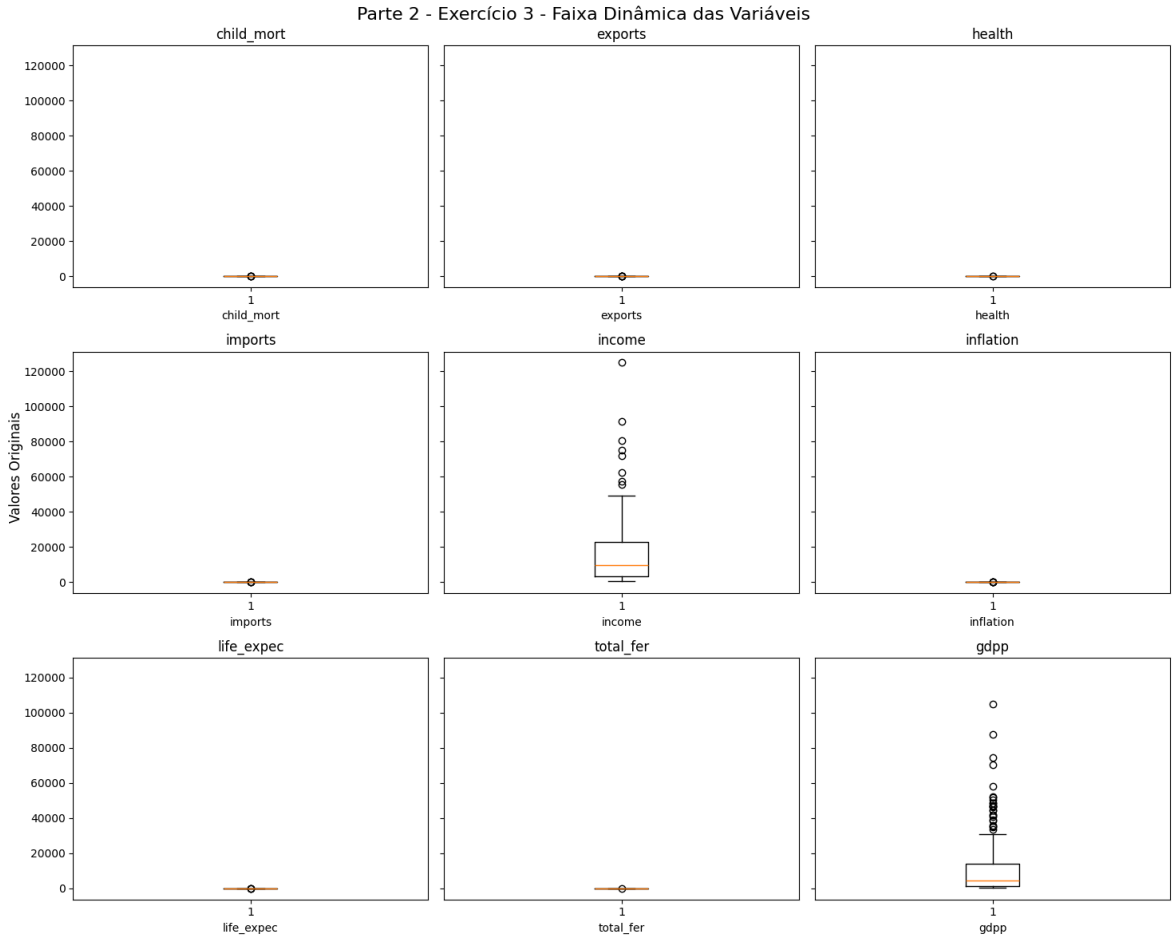
In [5]:
```
# Parte 2 - Exercício 3 - Mostre através de gráficos a faixa dinâmica das

n_linhas = 3
n_colunas = 3

fig, axes = plt.subplots(n_linhas, n_colunas, figsize=(15, 12), sharey=Tr
axes = axes.flatten()

for i, coluna in enumerate(colunas_de_dados):
    axes[i].boxplot(dataset[coluna])
    axes[i].set_title(coluna)
    axes[i].set_xlabel(coluna)
```

```
fig.supylabel('Valores Originais')
fig.suptitle('Parte 2 - Exercício 3 - Faixa Dinâmica das Variáveis', font
plt.tight_layout()
plt.show()
```



In [6]:
```
# Parte 2 - Exercício 3 - Analise os resultados mostrados. O que deve ser

print('\033[1mParte 2 - Exercício 3 - Resposta\033[0m\n')
print('Após a análise dos resultados da faixa dinâmica das variáveis, ver
```

Parte 2 - Exercício 3 - Resposta

Após a análise dos resultados da faixa dinâmica das variáveis, verifico qu
e devemos executar uma normalização de todas as colunas de dados na etapa
de pré-processamento.

In [7]:
```
# Parte 2 - Exercício 4 - Realize o pré-processamento adequado dos dados

# Mapeando nomes de países para números
pais_para_numero = {country: i for i, country in enumerate(dataset['count
# Mapeando números para nomes de países
numero_para_pais = {i: country for country, i in pais_para_numero.items()

print('\033[1mDicionário: pais_para_numero\033[0m')
print(pais_para_numero)
print('\033[1mDicionário: numero_para_pais\033[0m')
print(numero_para_pais)
```

Dicionário: pais_para_numero

{'Afghanistan': 0, 'Albania': 1, 'Algeria': 2, 'Angola': 3, 'Antigua and B
arbuda': 4, 'Argentina': 5, 'Armenia': 6, 'Australia': 7, 'Austria': 8, 'A
zerbaijan': 9, 'Bahamas': 10, 'Bahrain': 11, 'Bangladesh': 12, 'Barbados':
13, 'Belarus': 14, 'Belgium': 15, 'Belize': 16, 'Benin': 17, 'Bhutan': 18,
'Bolivia': 19, 'Bosnia and Herzegovina': 20, 'Botswana': 21, 'Brazil': 22,
'Brunei': 23, 'Bulgaria': 24, 'Burkina Faso': 25, 'Burundi': 26, 'Cambodi
a': 27, 'Cameroon': 28, 'Canada': 29, 'Cape Verde': 30, 'Central African R
epublic': 31, 'Chad': 32, 'Chile': 33, 'China': 34, 'Colombia': 35, 'Comor
os': 36, 'Congo, Dem. Rep.': 37, 'Congo, Rep.': 38, 'Costa Rica': 39, "Cot
e d'Ivoire": 40, 'Croatia': 41, 'Cyprus': 42, 'Czech Republic': 43, 'Denma
rk': 44, 'Dominican Republic': 45, 'Ecuador': 46, 'Egypt': 47, 'El Salvado
r': 48, 'Equatorial Guinea': 49, 'Eritrea': 50, 'Estonia': 51, 'Fiji': 52,
'Finland': 53, 'France': 54, 'Gabon': 55, 'Gambia': 56, 'Georgia': 57, 'Ge
rmany': 58, 'Ghana': 59, 'Greece': 60, 'Grenada': 61, 'Guatemala': 62, 'Gu
inea': 63, 'Guinea-Bissau': 64, 'Guyana': 65, 'Haiti': 66, 'Hungary': 67,
'Iceland': 68, 'India': 69, 'Indonesia': 70, 'Iran': 71, 'Iraq': 72, 'Irel
and': 73, 'Israel': 74, 'Italy': 75, 'Jamaica': 76, 'Japan': 77, 'Jordan':
78, 'Kazakhstan': 79, 'Kenya': 80, 'Kiribati': 81, 'Kuwait': 82, 'Kyrgyz R
epublic': 83, 'Lao': 84, 'Latvia': 85, 'Lebanon': 86, 'Lesotho': 87, 'Libe
ria': 88, 'Libya': 89, 'Lithuania': 90, 'Luxembourg': 91, 'Macedonia, FY
R': 92, 'Madagascar': 93, 'Malawi': 94, 'Malaysia': 95, 'Maldives': 96, 'M
ali': 97, 'Malta': 98, 'Mauritania': 99, 'Mauritius': 100, 'Micronesia, Fe
d. Sts.': 101, 'Moldova': 102, 'Mongolia': 103, 'Montenegro': 104, 'Morocc
o': 105, 'Mozambique': 106, 'Myanmar': 107, 'Namibia': 108, 'Nepal': 109,
'Netherlands': 110, 'New Zealand': 111, 'Niger': 112, 'Nigeria': 113, 'Nor
way': 114, 'Oman': 115, 'Pakistan': 116, 'Panama': 117, 'Paraguay': 118,
'Peru': 119, 'Philippines': 120, 'Poland': 121, 'Portugal': 122, 'Qatar':
123, 'Romania': 124, 'Russia': 125, 'Rwanda': 126, 'Samoa': 127, 'Saudi Ar
abia': 128, 'Senegal': 129, 'Serbia': 130, 'Seychelles': 131, 'Sierra Leon
e': 132, 'Singapore': 133, 'Slovak Republic': 134, 'Slovenia': 135, 'Solom
on Islands': 136, 'South Africa': 137, 'South Korea': 138, 'Spain': 139,
'Sri Lanka': 140, 'St. Vincent and the Grenadines': 141, 'Sudan': 142, 'Su
riname': 143, 'Sweden': 144, 'Switzerland': 145, 'Tajikistan': 146, 'Tanza
nia': 147, 'Thailand': 148, 'Timor-Leste': 149, 'Togo': 150, 'Tonga': 151,
'Tunisia': 152, 'Turkey': 153, 'Turkmenistan': 154, 'Uganda': 155, 'Ukrain
e': 156, 'United Arab Emirates': 157, 'United Kingdom': 158, 'United State
s': 159, 'Uruguay': 160, 'Uzbekistan': 161, 'Vanuatu': 162, 'Venezuela': 1
63, 'Vietnam': 164, 'Yemen': 165, 'Zambia': 166}

Dicionário: numero_para_pais

{0: 'Afghanistan', 1: 'Albania', 2: 'Algeria', 3: 'Angola', 4: 'Antigua an
d Barbuda', 5: 'Argentina', 6: 'Armenia', 7: 'Australia', 8: 'Austria', 9:
'Azerbaijan', 10: 'Bahamas', 11: 'Bahrain', 12: 'Bangladesh', 13: 'Barbado
s', 14: 'Belarus', 15: 'Belgium', 16: 'Belize', 17: 'Benin', 18: 'Bhutan',
19: 'Bolivia', 20: 'Bosnia and Herzegovina', 21: 'Botswana', 22: 'Brazil',
23: 'Brunei', 24: 'Bulgaria', 25: 'Burkina Faso', 26: 'Burundi', 27: 'Camb
odia', 28: 'Cameroon', 29: 'Canada', 30: 'Cape Verde', 31: 'Central Africa
n Republic', 32: 'Chad', 33: 'Chile', 34: 'China', 35: 'Colombia', 36: 'Co
moros', 37: 'Congo, Dem. Rep.', 38: 'Congo, Rep.', 39: 'Costa Rica', 40:
"Cote d'Ivoire", 41: 'Croatia', 42: 'Cyprus', 43: 'Czech Republic', 44: 'D
enmark', 45: 'Dominican Republic', 46: 'Ecuador', 47: 'Egypt', 48: 'El Sal
vador', 49: 'Equatorial Guinea', 50: 'Eritrea', 51: 'Estonia', 52: 'Fiji',
53: 'Finland', 54: 'France', 55: 'Gabon', 56: 'Gambia', 57: 'Georgia', 58:
'Germany', 59: 'Ghana', 60: 'Greece', 61: 'Grenada', 62: 'Guatemala', 63:
'Guinea', 64: 'Guinea-Bissau', 65: 'Guyana', 66: 'Haiti', 67: 'Hungary', 6
8: 'Iceland', 69: 'India', 70: 'Indonesia', 71: 'Iran', 72: 'Iraq', 73: 'I
reland', 74: 'Israel', 75: 'Italy', 76: 'Jamaica', 77: 'Japan', 78: 'Jorda
n', 79: 'Kazakhstan', 80: 'Kenya', 81: 'Kiribati', 82: 'Kuwait', 83: 'Kyrg
yz Republic', 84: 'Lao', 85: 'Latvia', 86: 'Lebanon', 87: 'Lesotho', 88:
'Liberia', 89: 'Libya', 90: 'Lithuania', 91: 'Luxembourg', 92: 'Macedonia,

```
FYR', 93: 'Madagascar', 94: 'Malawi', 95: 'Malaysia', 96: 'Maldives', 97:
'Mali', 98: 'Malta', 99: 'Mauritania', 100: 'Mauritius', 101: 'Micronesia,
Fed. Sts.', 102: 'Moldova', 103: 'Mongolia', 104: 'Montenegro', 105: 'Moro
cco', 106: 'Mozambique', 107: 'Myanmar', 108: 'Namibia', 109: 'Nepal', 11
0: 'Netherlands', 111: 'New Zealand', 112: 'Niger', 113: 'Nigeria', 114:
'Norway', 115: 'Oman', 116: 'Pakistan', 117: 'Panama', 118: 'Paraguay', 11
9: 'Peru', 120: 'Philippines', 121: 'Poland', 122: 'Portugal', 123: 'Qata
r', 124: 'Romania', 125: 'Russia', 126: 'Rwanda', 127: 'Samoa', 128: 'Saud
i Arabia', 129: 'Senegal', 130: 'Serbia', 131: 'Seychelles', 132: 'Sierra
Leone', 133: 'Singapore', 134: 'Slovak Republic', 135: 'Slovenia', 136: 'S
olomon Islands', 137: 'South Africa', 138: 'South Korea', 139: 'Spain', 14
0: 'Sri Lanka', 141: 'St. Vincent and the Grenadines', 142: 'Sudan', 143:
'Suriname', 144: 'Sweden', 145: 'Switzerland', 146: 'Tajikistan', 147: 'Ta
nzania', 148: 'Thailand', 149: 'Timor-Leste', 150: 'Togo', 151: 'Tonga', 1
52: 'Tunisia', 153: 'Turkey', 154: 'Turkmenistan', 155: 'Uganda', 156: 'Uk
raine', 157: 'United Arab Emirates', 158: 'United Kingdom', 159: 'United S
tates', 160: 'Uruguay', 161: 'Uzbekistan', 162: 'Vanuatu', 163: 'Venezuel
a', 164: 'Vietnam', 165: 'Yemen', 166: 'Zambia'}
```

In [8]:
```
# Parte 2 - Exercício 4 - Realize o pré-processamento adequado dos dados

# Realizando normalização de dados
scaler = MinMaxScaler()
dataset_normalizado = scaler.fit_transform(dataset[colunas_de_dados])
dataset_normalizado = pd.DataFrame(dataset_normalizado, columns=colunas_d

# Inserindo na primeira coluna do `dataset_normalizado` os códigos numéri
dataset_normalizado.insert(0, 'country_code', dataset['country'].map(pais

print('\033[1mDataset Normalizado com códigos numéricos para os países:\0
print(dataset_normalizado)
```

Dataset Normalizado com códigos numéricos para os países:

```
     country_code  child_mort   exports    health   imports    income  \
0               0    0.426485  0.049482  0.358608  0.257765  0.008047
1               1    0.068160  0.139531  0.294593  0.279037  0.074933
2               2    0.120253  0.191559  0.146675  0.180149  0.098809
3               3    0.566699  0.311125  0.064636  0.246266  0.042535
4               4    0.037488  0.227079  0.262275  0.338255  0.148652
..            ...         ...       ...       ...       ...       ...
162           162    0.129503  0.232582  0.213797  0.302609  0.018820
163           163    0.070594  0.142032  0.192666  0.100809  0.127750
164           164    0.100779  0.359651  0.312617  0.460715  0.031200
165           165    0.261441  0.149536  0.209447  0.197397  0.031120
166           166    0.391918  0.184556  0.253574  0.177275  0.021473

     inflation  life_expec  total_fer      gdpp
0     0.126144    0.475345   0.736593  0.003073
1     0.080399    0.871795   0.078864  0.036833
2     0.187691    0.875740   0.274448  0.040365
3     0.245911    0.552268   0.790221  0.031488
4     0.052213    0.881657   0.154574  0.114242
..         ...         ...        ...       ...
162   0.063118    0.609467   0.370662  0.026143
163   0.463081    0.854043   0.208202  0.126650
164   0.150725    0.808679   0.126183  0.010299
165   0.257000    0.698225   0.555205  0.010299
166   0.168284    0.392505   0.670347  0.011731

[167 rows x 10 columns]
```

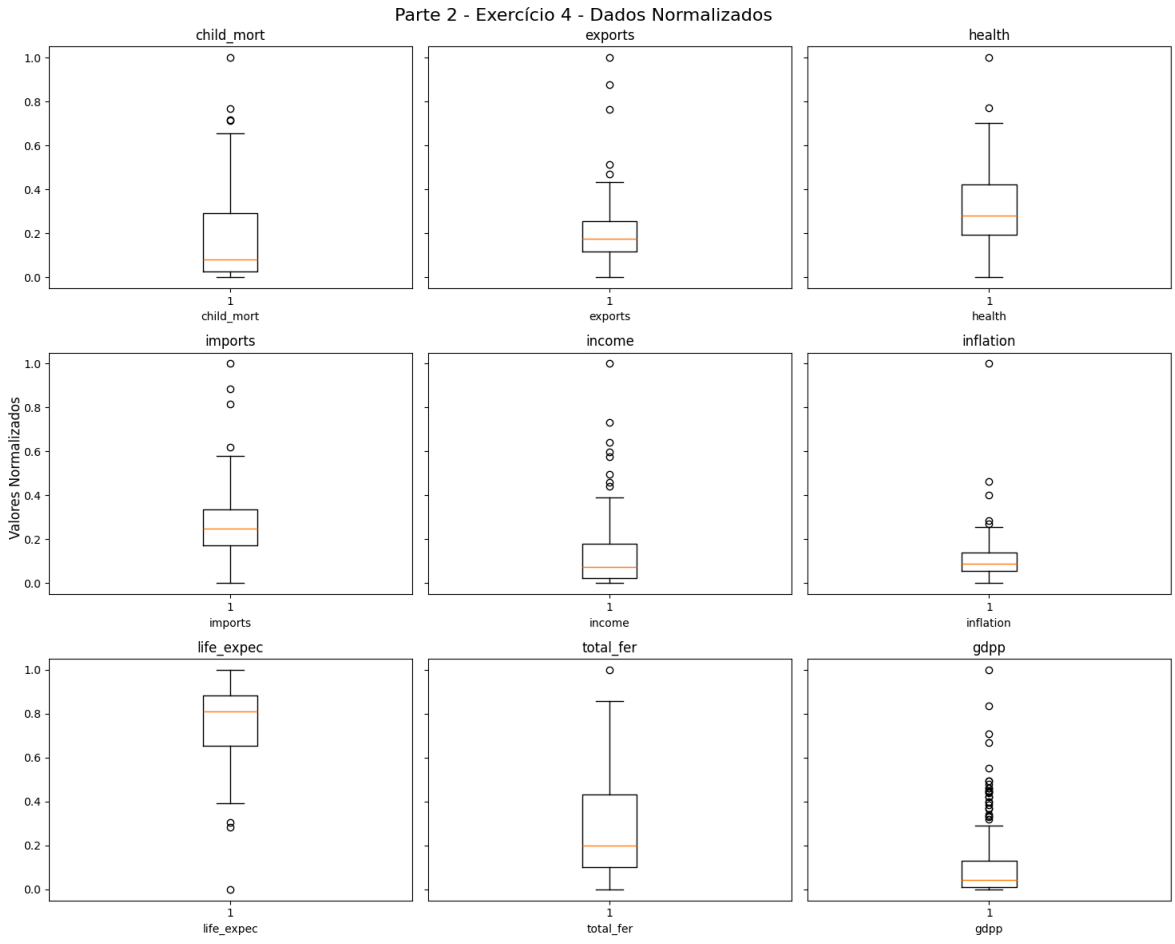
In [9]:
```
# Parte 2 - Exercício 4 - Realize o pré-processamento adequado dos dados

n_linhas = 3
n_colunas = 3

fig, axes = plt.subplots(n_linhas, n_colunas, figsize=(15, 12), sharey=Tr
axes = axes.flatten()

for i, coluna in enumerate(colunas_de_dados):
    axes[i].boxplot(dataset_normalizado[coluna])
    axes[i].set_title(coluna)
    axes[i].set_xlabel(coluna)

fig.supylabel('Valores Normalizados')
fig.suptitle('Parte 2 - Exercício 4 - Dados Normalizados', fontsize=16)
plt.tight_layout()
plt.show()
```



Parte 3 - Clusterização

Para os dados pré-processados da etapa anterior você irá:

1. Realizar o agrupamento dos países em 3 grupos distintos. Para tal, use:
   - a. K-Médias
   - b. Clusterização Hierárquica
2. Para os resultados, do K-Médias:
   - a. Interprete cada um dos clusters obtidos citando:
     - i. Qual a distribuição das dimensões em cada grupo

- ii. O país, de acordo com o algoritmo, melhor representa o seu agrupamento. Justifique

3. Para os resultados da Clusterização Hierárquica, apresente o dendograma e interprete os resultados
4. Compare os dois resultados, aponte as semelhanças e diferenças e interprete.

In [10]:
```
# Definição de número de clusters

n_clusters=3
```

In [11]:
```
# Parte 3 - Exercício 1.a - Agrupamento dos países em 3 grupos distintos

kmeans = KMeans(n_clusters=n_clusters,random_state=20)
dataset_normalizado['Cluster_Kmeans'] = kmeans.fit_predict(dataset_normal

print('\033[1mParte 3 - Exercício 1.a - Agrupamento dos países em 3 grupo
print(
    pd.DataFrame({
        'Country': dataset_normalizado['country_code'].map(numero_para_pa
        'Cluster_Kmeans': dataset_normalizado['Cluster_Kmeans']
    })
)
```

Parte 3 - Exercício 1.a - Agrupamento dos países em 3 grupos distintos usando K-Médias

```
                Country  Cluster_Kmeans
0           Afghanistan               1
1               Albania               2
2               Algeria               2
3                Angola               1
4    Antigua and Barbuda               2
..                   ...             ...
162              Vanuatu               2
163            Venezuela               2
164              Vietnam               2
165                Yemen               1
166               Zambia               1

[167 rows x 2 columns]
```

In [12]:
```
# Parte 3 - Exercício 1.b - Agrupamento dos países em 3 grupos distintos

data = dataset_normalizado[colunas_de_dados].values

clusterizacao_hierarquica = AgglomerativeClustering(n_clusters=n_clusters
dataset_normalizado['Cluster_Hierarquico'] = clusterizacao_hierarquica.fi

print('\033[1mParte 3 - Exercício 1.b - Agrupamento dos países em 3 grupo
print(
    pd.DataFrame({
        'Country': dataset_normalizado['country_code'].map(numero_para_pa
        'Cluster_Hierarquico': dataset_normalizado['Cluster_Hierarquico']
    })
)
```

Parte 3 - Exercício 1.b - Agrupamento dos países em 3 grupos distintos usando Clusterização Hierárquica

```
                Country  Cluster_Hierarquico
0           Afghanistan                    1
1               Albania                    2
2               Algeria                    2
3                Angola                    1
4    Antigua and Barbuda                    2
..                   ...                  ...
162              Vanuatu                    2
163            Venezuela                    2
164              Vietnam                    2
165                Yemen                    1
166               Zambia                    1

[167 rows x 2 columns]
```

In [13]:
```
# Parte 3 - Exercício 2.a.i - Para os resultados do K-Médias, interprete

distribuicao_clusters = dataset_normalizado.groupby('Cluster_Kmeans').mea
print('\033[1mParte 3 - Exercício 2.a.i(1) - Distribuição das dimensões e
print(distribuicao_clusters.iloc[:, 1:10])
```

Parte 3 - Exercício 2.a.i(1) - Distribuição das dimensões em cada grupo utilizando o K-Médias

```
                child_mort   exports    health   imports    income  inflat
ion  \
Cluster_Kmeans
0                 0.010883  0.289142  0.441962  0.291314  0.368192   0.062
890
1                 0.441503  0.145970  0.281447  0.248553  0.023376   0.150
698
2                 0.095659  0.203542  0.275648  0.271488  0.098919   0.108
466

                life_expec  total_fer      gdpp
Cluster_Kmeans
0                 0.952373   0.094684  0.416784
1                 0.538333   0.621485  0.013982
2                 0.799134   0.178759  0.063018
```

In [14]:
```
# Parte 3 - Exercício 2.a.i - Para os resultados do K-Médias, interprete

print('\033[1mParte 3 - Exercício 2.a.i - Para os resultados do K-Médias,

def classificar_cluster(valores):
    # Classificar os clusters automaticamente com base em dimensões-chave
    ## Países desenvolvidos possuem alta renda (income), alta expectativa
    if valores['income'] > 0.3 and valores['life_expec'] > 0.9 and valore
        return 'Países desenvolvidos'
    ## Países subdesenvolvidos possuem baixa renda (income), baixa expect
    elif valores['income'] < 0.1 and valores['life_expec'] < 0.6 and valo
        return 'Países subdesenvolvidos'
    ## Países em desenvolvimento possuem valores intermediários
    else:
        return 'Países em desenvolvimento'

def interpretar_clusters(distribuicao):
    interpretacoes = []
    for cluster, valores in distribuicao.iterrows():
```

```
        tipo_cluster = classificar_cluster(valores)
        interpretacao = (
            f'\033[1mCluster {cluster}: {tipo_cluster}\033[0m\n'
            f'- Mortalidade infantil: {valores['child_mort']:.4f}\n'
            f'- Expectativa de vida: {valores['life_expec']:.4f}\n'
            f'- Renda per capita: {valores['income']:.4f}\n'
            f'- Inflação: {valores['inflation']:.4f}\n'
            f'- PIB: {valores['gdpp']:.4f}\n'
        )
        interpretacoes.append(interpretacao)
    return interpretacoes

interpretacoes = interpretar_clusters(distribuicao_clusters)

for interpretacao in interpretacoes:
    print(interpretacao)
```

Parte 3 - Exercício 2.a.i - Para os resultados do K-Médias, interprete cad a um dos clusters obtidos citando: Qual a distribuição das dimensões em ca da grupo

Cluster 0: Países desenvolvidos
- Mortalidade infantil: 0.0109
- Expectativa de vida: 0.9524
- Renda per capita: 0.3682
- Inflação: 0.0629
- PIB: 0.4168

Cluster 1: Países subdesenvolvidos
- Mortalidade infantil: 0.4415
- Expectativa de vida: 0.5383
- Renda per capita: 0.0234
- Inflação: 0.1507
- PIB: 0.0140

Cluster 2: Países em desenvolvimento
- Mortalidade infantil: 0.0957
- Expectativa de vida: 0.7991
- Renda per capita: 0.0989
- Inflação: 0.1085
- PIB: 0.0630

In [15]:
```
# Parte 3 - Exercício 2.a.ii - Para os resultados do K-Médias, interprete

centroides = kmeans.cluster_centers_
distancias = cdist(dataset_normalizado[colunas_de_dados], centroides, met

representantes = {}
for cluster in range(n_clusters):
    indices_cluster = dataset_normalizado['Cluster_Kmeans'] == cluster
    distancias_cluster = distancias[indices_cluster, cluster]
    indice_mais_proximo = np.argmin(distancias_cluster)
    pais_mais_proximo = dataset_normalizado.loc[indices_cluster].iloc[ind
    representantes[cluster] = numero_para_pais[pais_mais_proximo]

print('\033[1mParte 3 - Exercício 2.a.ii(1) - Distribuição das dimensões
print('\033[1mPaíses que melhor representam cada agrupamento:\033[0m\n')
for cluster, pais in representantes.items():
    print(f'\033[1mCluster {cluster}:\033[0m {pais}')
```

Parte 3 – Exercício 2.a.ii(1) – Distribuição das dimensões em cada grupo utilizando o K-Médias
Países que melhor representam cada agrupamento:

Cluster 0: Iceland
Cluster 1: Guinea
Cluster 2: Suriname

In [16]:
```
# Parte 3 – Exercício 2.a.ii – Para os resultados do K-Médias, interprete

print('\033[1mParte 3 – Exercício 2.a.ii(2) – Distribuição das dimensões

def justificar_representantes(representantes, centroides, dataset, coluna
    justificativas = []

    for cluster, pais in representantes.items():
        country_code = list(numero_para_pais.keys())[list(numero_para_pai
        valores_pais = dataset[dataset['country_code'] == country_code][c
        valores_centroide = centroides[cluster]
        diferencas = abs(valores_pais - valores_centroide)

        menores_indices = diferencas.argsort()[:3]
        menores_colunas = [colunas[i] for i in menores_indices]
        menores_diferencas = [diferencas.iloc[i] for i in menores_indices

        justificativa = (
            f'\033[1mCluster {cluster} – Representante: {pais}\033[0m\n'
            f'Este país foi escolhido porque possui valores próximos ao c
            f'As 3 dimensões com menor diferença absoluta entre o país \0
        )

        for coluna, diferenca in zip(menores_colunas, menores_diferencas)
            justificativa += f'– {coluna}: {diferenca:.4f}\n'

        justificativas.append(justificativa)

    return justificativas

justificativas = justificar_representantes(representantes, centroides, da

for justificativa in justificativas:
    print(justificativa)
```

Parte 3 – Exercício 2.a.ii(2) – Distribuição das dimensões em cada grupo utilizando o K-Médias – Justificativas

Cluster 0 – Representante: Iceland
Este país foi escolhido porque possui valores próximos ao centróide do cluster.
As 3 dimensões com menor diferença absoluta entre o país **Iceland** e o centróide do cluster são:
- child_mort: 0.0109
- gdpp: 0.0191
- exports: 0.0225

Cluster 1 – Representante: Guinea
Este país foi escolhido porque possui valores próximos ao centróide do cluster.
As 3 dimensões com menor diferença absoluta entre o país **Guinea** e o centróide do cluster são:
- imports: 0.0006
- exports: 0.0051
- gdpp: 0.0100

Cluster 2 – Representante: Suriname
Este país foi escolhido porque possui valores próximos ao centróide do cluster.
As 3 dimensões com menor diferença absoluta entre o país **Suriname** e o centróide do cluster são:
- inflation: 0.0030
- child_mort: 0.0090
- income: 0.0103

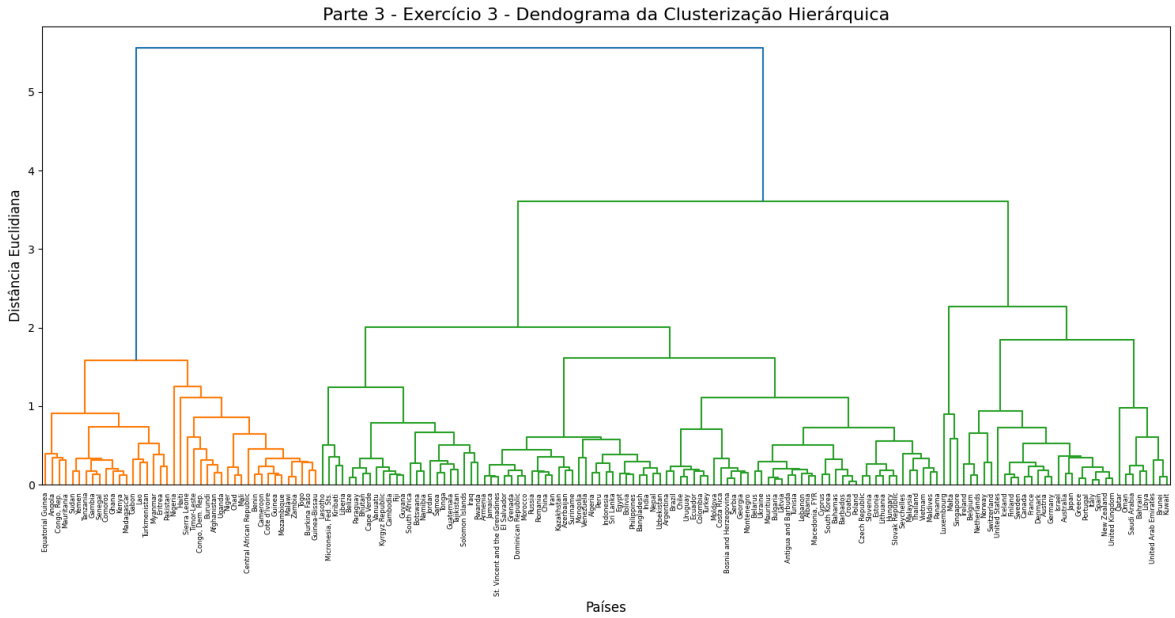
In [17]:
```
# Parte 3 – Exercício 3 – Para os resultados da Clusterização Hierárquica

# Criar a matriz de linkagem para o dendrograma usando o método 'ward'
linkage_matrix = linkage(data, method='ward')

# Função para plotar o dendrograma
plt.figure(figsize=(15, 8))
plt.title('Parte 3 – Exercício 3 – Dendograma da Clusterização Hierárquic
plt.xlabel('Países', fontsize=12)
plt.ylabel('Distância Euclidiana', fontsize=12)

dendrogram(
    linkage_matrix,
    labels=dataset_normalizado['country_code'].map(numero_para_pais).valu
)

plt.tight_layout()
plt.show()
```



In [18]:
```
# Parte 3 – Exercício 3 – Para os resultados da Clusterização Hierárquica

# Listar os países em cada cluster hierárquico
for cluster in sorted(dataset_normalizado['Cluster_Hierarquico'].unique()):
    print(f'Cluster {cluster}:')
    print(dataset_normalizado[dataset_normalizado['Cluster_Hierarquico'] == cluster]
          .map(numero_para_pais)
          .tolist())
    print()
```

```
Cluster 0:
['Australia', 'Austria', 'Bahrain', 'Belgium', 'Brunei', 'Canada', 'Denmar
k', 'Finland', 'France', 'Germany', 'Greece', 'Iceland', 'Ireland', 'Israe
l', 'Italy', 'Japan', 'Kuwait', 'Libya', 'Luxembourg', 'Malta', 'Netherlan
ds', 'New Zealand', 'Norway', 'Oman', 'Portugal', 'Qatar', 'Saudi Arabia',
'Singapore', 'Spain', 'Sweden', 'Switzerland', 'United Arab Emirates', 'Un
ited Kingdom', 'United States']

Cluster 1:
['Afghanistan', 'Angola', 'Benin', 'Burkina Faso', 'Burundi', 'Cameroon',
'Central African Republic', 'Chad', 'Comoros', 'Congo, Dem. Rep.', 'Congo,
Rep.', "Cote d'Ivoire", 'Equatorial Guinea', 'Eritrea', 'Gabon', 'Gambia',
'Ghana', 'Guinea', 'Guinea-Bissau', 'Haiti', 'Kenya', 'Lao', 'Madagascar',
'Malawi', 'Mali', 'Mauritania', 'Mozambique', 'Myanmar', 'Niger', 'Nigeri
a', 'Pakistan', 'Senegal', 'Sierra Leone', 'Sudan', 'Tanzania', 'Timor-Les
te', 'Togo', 'Turkmenistan', 'Uganda', 'Yemen', 'Zambia']

Cluster 2:
['Albania', 'Algeria', 'Antigua and Barbuda', 'Argentina', 'Armenia', 'Aze
rbaijan', 'Bahamas', 'Bangladesh', 'Barbados', 'Belarus', 'Belize', 'Bhuta
n', 'Bolivia', 'Bosnia and Herzegovina', 'Botswana', 'Brazil', 'Bulgaria',
'Cambodia', 'Cape Verde', 'Chile', 'China', 'Colombia', 'Costa Rica', 'Cro
atia', 'Cyprus', 'Czech Republic', 'Dominican Republic', 'Ecuador', 'Egyp
t', 'El Salvador', 'Estonia', 'Fiji', 'Georgia', 'Grenada', 'Guatemala',
'Guyana', 'Hungary', 'India', 'Indonesia', 'Iran', 'Iraq', 'Jamaica', 'Jor
dan', 'Kazakhstan', 'Kiribati', 'Kyrgyz Republic', 'Latvia', 'Lebanon', 'L
esotho', 'Liberia', 'Lithuania', 'Macedonia, FYR', 'Malaysia', 'Maldives',
'Mauritius', 'Micronesia, Fed. Sts.', 'Moldova', 'Mongolia', 'Montenegro',
'Morocco', 'Namibia', 'Nepal', 'Panama', 'Paraguay', 'Peru', 'Philippine
s', 'Poland', 'Romania', 'Russia', 'Rwanda', 'Samoa', 'Serbia', 'Seychelle
s', 'Slovak Republic', 'Slovenia', 'Solomon Islands', 'South Africa', 'Sou
th Korea', 'Sri Lanka', 'St. Vincent and the Grenadines', 'Suriname', 'Taj
ikistan', 'Thailand', 'Tonga', 'Tunisia', 'Turkey', 'Ukraine', 'Uruguay',
'Uzbekistan', 'Vanuatu', 'Venezuela', 'Vietnam']
```

In [19]:
```
# Parte 3 - Exercício 3 - Para os resultados da Clusterização Hierárquica

print('## Justificativa dos Clusters - Dendograma da Clusterização Hierár

## Cluster 1
print('\n\033[1m### Cluster 1 (Laranja - Subdesenvolvidos)\033[0m')
print('- \033[1mPaíses incluídos\033[0m: Camarões, Angola, Serra Leoa, Co
print('- \033[1mJustificação\033[0m:')
print('  - Esses países compartilham baixos indicadores socioeconômicos.'
print('  - \033[1mCaracterísticas principais\033[0m:')
print('    - \033[1mRenda per capita\033[0m: Baixa.')
print('    - \033[1mMortalidade infantil\033[0m: Alta.')
print('    - \033[1mExpectativa de vida\033[0m: Baixa.')

## Cluster 2
print('\n\033[1m### Cluster 2 (Verde à esquerda - Em Desenvolvimento)\033
print('- \033[1mPaíses incluídos\033[0m: Micronésia, Brasil, Paraguai, en
print('- \033[1mJustificação\033[0m:')
print('  - Estes países possuem características intermediárias, tanto soc
print('  - \033[1mCaracterísticas principais\033[0m:')
print('    - \033[1mRenda per capita\033[0m: Moderada.')
print('    - \033[1mMortalidade infantil\033[0m: Moderada.')
print('    - \033[1mExpectativa de vida\033[0m: Moderada.')
```

```
## Cluster 3
print('\n\033[1m### Cluster 3 (Verde à direita - Desenvolvidos)\033[0m')
print('- \033[1mPaíses incluídos\033[0m: Japão, Estados Unidos, Alemanha,
print('- \033[1mJustificação\033[0m:')
print('  - Esses países têm indicadores socioeconômicos elevados, refleti
print('  - \033[1mCaracterísticas principais\033[0m:')
print('    - \033[1mRenda per capita\033[0m: Alta.')
print('    - \033[1mMortalidade infantil\033[0m: Baixa.')
print('    - \033[1mExpectativa de vida\033[0m: Alta.')

## Observações gerais
print('\n\033[1m### Observações gerais\033[0m')
print('- A \033[1maltura das uniões\033[0m no dendograma reflete as difer
print('  - Clusters unidos em alturas maiores são mais heterogêneos.')
print('  - Clusters unidos em alturas menores são mais homogêneos.')
print('- A separação em 3 grandes clusters foi realizada com base em uma
```

Justificativa dos Clusters - Dendograma da Clusterização Hierárquica

Cluster 1 (Laranja - Subdesenvolvidos)
- **Países incluídos**: Camarões, Angola, Serra Leoa, Congo, entre outros.
- **Justificação**:
  - Esses países compartilham baixos indicadores socioeconômicos.
  - **Características principais**:
    - **Renda per capita**: Baixa.
    - **Mortalidade infantil**: Alta.
    - **Expectativa de vida**: Baixa.

Cluster 2 (Verde à esquerda - Em Desenvolvimento)
- **Países incluídos**: Micronésia, Brasil, Paraguai, entre outros.
- **Justificação**:
  - Estes países possuem características intermediárias, tanto sociais qua
nto econômicas.
  - **Características principais**:
    - **Renda per capita**: Moderada.
    - **Mortalidade infantil**: Moderada.
    - **Expectativa de vida**: Moderada.

Cluster 3 (Verde à direita - Desenvolvidos)
- **Países incluídos**: Japão, Estados Unidos, Alemanha, entre outros.
- **Justificação**:
  - Esses países têm indicadores socioeconômicos elevados, refletindo alto
desenvolvimento.
  - **Características principais**:
    - **Renda per capita**: Alta.
    - **Mortalidade infantil**: Baixa.
    - **Expectativa de vida**: Alta.

Observações gerais
- A **altura das uniões** no dendograma reflete as diferenças entre os grupos:
  - Clusters unidos em alturas maiores são mais heterogêneos.
  - Clusters unidos em alturas menores são mais homogêneos.
- A separação em 3 grandes clusters foi realizada com base em uma altura d
e corte de aproximadamente **2.5** no eixo Y.

In [20]:
```
# Parte 3 - Exercício 4 - Compare os dois resultados, aponte as semelhanç
# OUTPUT: parte3_exercicio4_comparacao_clusters.csv

comparacao_clusters = dataset_normalizado[['country_code', 'Cluster_Kmean
comparacao_clusters.loc[:, 'Country'] = comparacao_clusters['country_code
```

```
comparacao_clusters = comparacao_clusters[['Country', 'Cluster_Kmeans', '

# Adicionar uma coluna indicando se os clusters coincidem
comparacao_clusters['Mesmo_Cluster'] = comparacao_clusters['Cluster_Kmean

# Exibir os primeiros 20 resultados
print('\033[1mParte 3 - Exercício 4 - Tabela comparativa\033[0m\n')
print(comparacao_clusters.head(20))

# Opcional: salvar a tabela de comparação em um arquivo CSV
comparacao_clusters.to_csv('parte3_exercicio4_comparacao_clusters.csv', i
print('Tabela de comparação salva como "parte3_exercicio4_comparacao_clus
```

Parte 3 - Exercício 4 - Tabela comparativa

```
                Country  Cluster_Kmeans  Cluster_Hierarquico  Mesmo_Cluste
r
0           Afghanistan               1                    1           Tru
e
1               Albania               2                    2           Tru
e
2               Algeria               2                    2           Tru
e
3                Angola               1                    1           Tru
e
4   Antigua and Barbuda               2                    2           Tru
e
5             Argentina               2                    2           Tru
e
6               Armenia               2                    2           Tru
e
7             Australia               0                    0           Tru
e
8               Austria               0                    0           Tru
e
9            Azerbaijan               2                    2           Tru
e
10              Bahamas               2                    2           Tru
e
11              Bahrain               2                    0          Fals
e
12           Bangladesh               2                    2           Tru
e
13             Barbados               2                    2           Tru
e
14              Belarus               2                    2           Tru
e
15              Belgium               0                    0           Tru
e
16               Belize               2                    2           Tru
e
17                Benin               1                    1           Tru
e
18               Bhutan               2                    2           Tru
e
19              Bolivia               2                    2           Tru
e
Tabela de comparação salva como "parte3_exercicio4_comparacao_clusters.cs
v"
```

In [21]:
```
# Parte 3 - Exercício 4 - Compare os dois resultados, aponte as semelhanç

# Calcular estatísticas para a interpretação
total_paises = len(comparacao_clusters)
contagem_mesmo_cluster = comparacao_clusters['Mesmo_Cluster'].sum()
proporcao_mesmo_cluster = contagem_mesmo_cluster / total_paises

# Países com clusters diferentes
clusters_diferentes = comparacao_clusters[comparacao_clusters['Mesmo_Clus

# Exibir a interpretação
print('\033[1mParte 3 - Exercício 4 - Diferenças\033[0m\n')
print(f'Total de países analisados: {total_paises}')
print(f'Número de países no mesmo cluster nos dois métodos: {contagem_mes
print(f'Proporção de países no mesmo cluster: {proporcao_mesmo_cluster:.2

if len(clusters_diferentes) > 0:
    print('Países classificados em clusters diferentes:')
    print(clusters_diferentes[['Country', 'Cluster_Kmeans', 'Cluster_Hier
else:
    print('Todos os países foram classificados no mesmo cluster nos dois
```

Parte 3 - Exercício 4 - Diferenças

Total de países analisados: 167
Número de países no mesmo cluster nos dois métodos: 150
Proporção de países no mesmo cluster: 89.82%

Países classificados em clusters diferentes:

Country	Cluster_Kmeans	Cluster_Hierarquico
Bahrain	2	0
Cyprus	0	2
Czech Republic	0	2
Iraq	1	2
Kiribati	1	2
Lesotho	1	2
Liberia	1	2
Libya	2	0
Myanmar	2	1
Namibia	1	2
Oman	2	0
Rwanda	1	2
Saudi Arabia	2	0
Slovenia	0	2
Solomon Islands	1	2
South Korea	0	2
Turkmenistan	2	1

In [22]:
```
# Parte 3 - Exercício 4 - Compare os dois resultados, aponte as semelhanç
# OUTPUT: parte3_exercicio4justificacao_diferencas_clusters.csv

# Selecionar os países classificados de forma diferente
diferencas_clusters = comparacao_clusters[comparacao_clusters['Mesmo_Clus

# Obter os dados completos desses países usando a coluna 'Country'
diferencas_dados = dataset_normalizado[dataset_normalizado['country_code'

# Adicionar os clusters K-Means e Hierárquico para comparação
diferencas_dados = diferencas_dados[['country_code'] + colunas_de_dados +
diferencas_dados['Country'] = diferencas_dados['country_code'].map(numero
```

```
# Reorganizar as colunas para melhor visualização
diferencas_dados = diferencas_dados[['Country', 'Cluster_Kmeans', 'Cluste

# Analisar as diferenças entre os clusters
print('\033[1mParte 3 - Exercício 4 - Interpretação\033[0m')
for _, row in diferencas_dados.iterrows():
    print(f'\033[1m\nPaís: {row["Country"]}\033[0m')
    print(f'Cluster K-Means: {row["Cluster_Kmeans"]} | Cluster Hierárquic
    print('Características principais:')
    for coluna in colunas_de_dados:
        print(f'- {coluna}: {row[coluna]:.4f}')
    print('Justificação:')
    if row['Cluster_Kmeans'] != row['Cluster_Hierarquico']:
        print('-> Diferença de classificação: Este país apresenta caracte
              ' o que leva os dois métodos a classificá-lo de maneira dis
    print('-' * 50)

# Salvar as diferenças em um arquivo CSV para análise adicional
diferencas_dados.to_csv('parte3_exercicio4_justificacao_diferencas_cluste
print('\nArquivo \'parte3_exercicio4_justificacao_diferencas_clusters.csv
```

Parte 3 - Exercício 4 - Interpretação

País: Bahrain
Cluster K-Means: 2 | Cluster Hierárquico: 0
Características principais:
- child_mort: 0.0292
- exports: 0.3471
- health: 0.1964
- imports: 0.2923
- income: 0.3255
- inflation: 0.1077
- life_expec: 0.8659
- total_fer: 0.1593
- gdpp: 0.1954
Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

País: Cyprus
Cluster K-Means: 0 | Cluster Hierárquico: 2
Características principais:
- child_mort: 0.0049
- exports: 0.2506
- health: 0.2585
- imports: 0.3302
- income: 0.2676
- inflation: 0.0575
- life_expec: 0.9428
- total_fer: 0.0426
- gdpp: 0.2918
Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

País: Czech Republic
Cluster K-Means: 0 | Cluster Hierárquico: 2
Características principais:
- child_mort: 0.0039
- exports: 0.3296
- health: 0.3773
- imports: 0.3613
- income: 0.2226
- inflation: 0.0257
- life_expec: 0.8955
- total_fer: 0.0568
- gdpp: 0.1868
Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

País: Iraq
Cluster K-Means: 1 | Cluster Hierárquico: 2
Características principais:
- child_mort: 0.1670

- exports: 0.1966
- health: 0.4102
- imports: 0.1957
- income: 0.0972
- inflation: 0.1923
- life_expec: 0.6923
- total_fer: 0.5379
- gdpp: 0.0407

Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

País: Kiribati
Cluster K-Means: 1 | Cluster Hierárquico: 2
Características principais:
- child_mort: 0.2926
- exports: 0.0660
- health: 0.5898
- imports: 0.4590
- income: 0.0090
- inflation: 0.0530
- life_expec: 0.5641
- total_fer: 0.4243
- gdpp: 0.0120

Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

País: Lesotho
Cluster K-Means: 1 | Cluster Hierárquico: 2
Características principais:
- child_mort: 0.4727
- exports: 0.1966
- health: 0.5774
- imports: 0.5803
- income: 0.0142
- inflation: 0.0773
- life_expec: 0.2840
- total_fer: 0.3391
- gdpp: 0.0090

Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

País: Liberia
Cluster K-Means: 1 | Cluster Hierárquico: 2
Características principais:
- child_mort: 0.4221
- exports: 0.0950
- health: 0.6209
- imports: 0.5320
- income: 0.0007
- inflation: 0.0895
- life_expec: 0.5661

- total_fer: 0.6104
- gdpp: 0.0009
Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

País: Libya
Cluster K-Means: 2 | Cluster Hierárquico: 0
Características principais:
- child_mort: 0.0682
- exports: 0.3276
- health: 0.1287
- imports: 0.2417
- income: 0.2331
- inflation: 0.1701
- life_expec: 0.8679
- total_fer: 0.1987
- gdpp: 0.1133
Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

País: Myanmar
Cluster K-Means: 2 | Cluster Hierárquico: 1
Características principais:
- child_mort: 0.3009
- exports: 0.0000
- health: 0.0099
- imports: 0.0000
- income: 0.0250
- inflation: 0.1040
- life_expec: 0.6844
- total_fer: 0.1987
- gdpp: 0.0072
Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

País: Namibia
Cluster K-Means: 1 | Cluster Hierárquico: 2
Características principais:
- child_mort: 0.2600
- exports: 0.2386
- health: 0.3089
- imports: 0.3486
- income: 0.0631
- inflation: 0.0718
- life_expec: 0.5227
- total_fer: 0.3864
- gdpp: 0.0473
Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

```
--------------------------------------------------
```

País: Oman
Cluster K-Means: 2 | Cluster Hierárquico: 0
Características principais:
- child_mort: 0.0443
- exports: 0.3281
- health: 0.0597
- imports: 0.2365
- income: 0.3593
- inflation: 0.1831
- life_expec: 0.8679
- total_fer: 0.2760
- gdpp: 0.1820
Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

```
--------------------------------------------------
```

País: Rwanda
Cluster K-Means: 1 | Cluster Hierárquico: 2
Características principais:
- child_mort: 0.2970
- exports: 0.0595
- health: 0.5401
- imports: 0.1721
- income: 0.0060
- inflation: 0.0630
- life_expec: 0.6410
- total_fer: 0.5300
- gdpp: 0.0032
Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

```
--------------------------------------------------
```

País: Saudi Arabia
Cluster K-Means: 2 | Cluster Hierárquico: 0
Características principais:
- child_mort: 0.0638
- exports: 0.2476
- health: 0.1541
- imports: 0.1893
- income: 0.3601
- inflation: 0.1979
- life_expec: 0.8481
- total_fer: 0.2855
- gdpp: 0.1820
Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

```
--------------------------------------------------
```

País: Slovenia
Cluster K-Means: 0 | Cluster Hierárquico: 2
Características principais:
- child_mort: 0.0029

- exports: 0.3211
- health: 0.4723
- imports: 0.3613
- income: 0.2258
- inflation: 0.0298
- life_expec: 0.9349
- total_fer: 0.0662
- gdpp: 0.2211

Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

País: Solomon Islands
Cluster K-Means: 1 | Cluster Hierárquico: 2
Características principais:
- child_mort: 0.1241
- exports: 0.2461
- health: 0.4189
- imports: 0.4665
- income: 0.0094
- inflation: 0.1018
- life_expec: 0.5838
- total_fer: 0.4874
- gdpp: 0.0101

Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

País: South Korea
Cluster K-Means: 0 | Cluster Hierárquico: 2
Características principais:
- child_mort: 0.0073
- exports: 0.2466
- health: 0.3182
- imports: 0.2652
- income: 0.2395
- inflation: 0.0681
- life_expec: 0.9467
- total_fer: 0.0126
- gdpp: 0.2087

Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.

País: Turkmenistan
Cluster K-Means: 2 | Cluster Hierárquico: 1
Características principais:
- child_mort: 0.2892
- exports: 0.3812
- health: 0.0429
- imports: 0.2555
- income: 0.0750
- inflation: 0.0603
- life_expec: 0.7061

```
- total_fer: 0.2650
- gdpp: 0.0402
Justificação:
-> Diferença de classificação: Este país apresenta características que est
ão 'na fronteira' entre os clusters, o que leva os dois métodos a classifi
cá-lo de maneira distinta com base nas distâncias e abordagens.
--------------------------------------------------

Arquivo 'parte3_exercicio4_justificacao_diferencas_clusters.csv' criado co
m os dados detalhados.
```

Parte 4 - Escolha de Algoritmos

1. Escreva em tópicos as etapas do algoritmo de K-médias até sua convergência.
2. O algoritmo de K-médias converge até encontrar os centróides que melhor descrevem os clusters encontrados (até o deslocamento entre as interações dos centróides ser mínimo). Lembrando que o centróide é o baricentro do cluster em questão e não representa, em via de regra, um dado existente na base. Refaça o algoritmo apresentado na questão 1 a fim de garantir que o cluster seja representado pelo dado mais próximo ao seu baricentro em todas as iterações do algoritmo.
    - Obs: nesse novo algoritmo, o dado escolhido será chamado medóide.
3. O algoritmo de K-médias é sensível a outliers nos dados. Explique.
4. Por que o algoritmo de DBScan é mais robusto à presença de outliers?

In [23]:
```
# Parte 4 - Exercício 1 - Escreva em tópicos as etapas do algoritmo de K-

print('\033[1mParte 4 - Exercício 1 - Escreva em tópicos as etapas do alg

print('\n\033[1mEtapa 1: Escolha do número de grupos (clusters).\033[0m')
print('* Decisão do número de grupos para dividir os dados. Esse número é

print('\n\033[1mEtapa 2: Posicionamento inicial dos centros (centróides).
print('* Os centróides iniciais são escolhidos automaticamente pelo algor
print('* No código, usei "KMeans(n_clusters=n_clusters, random_state=20)"

print('\n\033[1mEtapa 3: Agrupamento inicial dos pontos.\033[0m')
print('* Para cada ponto nos dados:')
print('   - O algoritmo calcula qual centróide está mais próximo.')
print('   - O ponto é então atribuído ao grupo (cluster) correspondente.'

print('\n\033[1mEtapa 4: Ajuste dos centros dos grupos.\033[0m')
print('* Após a atribuição inicial:')
print('   - O algoritmo calcula o "novo centro" de cada grupo.')
print('   - Este novo centro é a posição média de todos os pontos atribuí
print('   - Os centróides são ajustados para essas novas posições.')

print('\n\033[1mEtapa 5: Reagrupamento dos pontos.\033[0m')
print('* Com os centróides ajustados:')
print('   - Cada ponto é reavaliado para verificar qual centróide está ma
print('   - Se algum ponto mudar de grupo, o agrupamento é atualizado.')

print('\n\033[1mEtapa 6: Repetição até estabilizar.\033[0m')
print('* O processo de ajuste e reagrupamento continua até que:')
print('   - Nenhum ponto mude de grupo.')
```

```
print('   - Ou o algoritmo atinja um número máximo de iterações.')

print('\n\033[1mEtapa 7: Resultados finais.\033[0m')
print('* No final, o algoritmo retorna:')
print('   - Todos os dados divididos em k grupos.')
print('   - Os centróides finais, que representam o "coração" de cada gru
```

Parte 4 - Exercício 1 - Escreva em tópicos as etapas do algoritmo de K-méd ias até sua convergência.o

Etapa 1: Escolha do número de grupos (clusters).
* Decisão do número de grupos para dividir os dados. Esse número é chamado de k.

Etapa 2: Posicionamento inicial dos centros (centróides).
* Os centróides iniciais são escolhidos automaticamente pelo algoritmo de K-Means.
* No código, usei "KMeans(n_clusters=n_clusters, random_state=20)" para de finir o número de clusters e garantir reprodutibilidade.

Etapa 3: Agrupamento inicial dos pontos.
* Para cada ponto nos dados:
   - O algoritmo calcula qual centróide está mais próximo.
   - O ponto é então atribuído ao grupo (cluster) correspondente.

Etapa 4: Ajuste dos centros dos grupos.
* Após a atribuição inicial:
   - O algoritmo calcula o "novo centro" de cada grupo.
   - Este novo centro é a posição média de todos os pontos atribuídos ao g rupo.
   - Os centróides são ajustados para essas novas posições.

Etapa 5: Reagrupamento dos pontos.
* Com os centróides ajustados:
   - Cada ponto é reavaliado para verificar qual centróide está mais próxi mo.
   - Se algum ponto mudar de grupo, o agrupamento é atualizado.

Etapa 6: Repetição até estabilizar.
* O processo de ajuste e reagrupamento continua até que:
   - Nenhum ponto mude de grupo.
   - Ou o algoritmo atinja um número máximo de iterações.

Etapa 7: Resultados finais.
* No final, o algoritmo retorna:
   - Todos os dados divididos em k grupos.
   - Os centróides finais, que representam o "coração" de cada grupo.

In [24]:
```
# Parte 4 - Exercício 2 - O algoritmo de K-médias converge até encontrar
# Lembrando que o centróide é o baricentro do cluster em questão e não re
# Refaça o algoritmo apresentado na questão 1 a fim de garantir que o clu
# Obs: nesse novo algoritmo, o dado escolhido será chamado medóide.

# Inicializar os medóides com índices aleatórios
np.random.seed(20)
indices_iniciais = np.random.choice(len(dataset_normalizado), size=n_clus
medoids = dataset_normalizado[colunas_de_dados].iloc[indices_iniciais].va

# Função para calcular clusters e custo
def atribuir_clusters_e_calcular_custo(data, medoids):
```

```
    distancias = cdist(data, medoids, metric='euclidean')  # Distâncias E
    clusters = np.argmin(distancias, axis=1)  # Atribuir ao medóide mais
    custo = np.sum(np.min(distancias, axis=1))  # Soma das distâncias mín
    return clusters, custo

# Iterar até convergência
medoide_anterior = None
iteracao_kmedoid = 0
while not np.array_equal(medoide_anterior, medoids):
    iteracao_kmedoid += 1
    medoide_anterior = medoids.copy()
    # Atribuir clusters e calcular o custo
    dataset_normalizado['Cluster_Kmedoids'], _ = atribuir_clusters_e_calc
        dataset_normalizado[colunas_de_dados].values, medoids
    )
    # Atualizar medóides para o ponto mais central do cluster
    for cluster in range(n_clusters):
        cluster_points = dataset_normalizado[dataset_normalizado['Cluster
        medoid_index = np.argmin(np.sum(cdist(cluster_points, cluster_poi
        medoids[cluster] = cluster_points[medoid_index]

# Exibir os resultados
print('\033[1mParte 4 - Exercício 2 - Utilização de K-medoids\033[0m\n')

print(f'Convergência alcançada após \033[1m{iteracao_kmedoid}\033[0m iter

print(
    pd.DataFrame({
        'Country': dataset_normalizado['country_code'].map(numero_para_pa
        'Cluster_Kmedoids': dataset_normalizado['Cluster_Kmedoids']
    })
)
```

Parte 4 - Exercício 2 - Utilização de K-medoids

Convergência alcançada após **3** iterações

	Country	Cluster_Kmedoids
0	Afghanistan	0
1	Albania	2
2	Algeria	2
3	Angola	0
4	Antigua and Barbuda	2
..	...	...
162	Vanuatu	1
163	Venezuela	2
164	Vietnam	2
165	Yemen	1
166	Zambia	0

[167 rows x 2 columns]

In [25]:
```
# Parte 4 - Exercício 3 - O algoritmo de K-médias é sensível a outliers n

print('\033[1mParte 4 - Exercício 3 - O algoritmo de K-médias é sensível

print('\033[1mPor que o K-Means é sensível a outliers?\033[0m')
print('* O algoritmo calcula os centróides como a média dos pontos no clu
print('* Outliers, que são valores muito distantes, podem deslocar o cent
```

```
print('\n\033[1mImpacto nos agrupamentos:\033[0m')
print('* Os clusters podem ficar distorcidos, não refletindo bem os dados
print('* Em alguns casos, os outliers podem até formar clusters próprios,

print('\n\033[1mComo evitar o problema?\033[0m')
print('\033[1m1.\033[0m Remover ou tratar os outliers antes de rodar o al
print('\033[1m2.\033[0m Normalizar os dados para reduzir o impacto de val
print('\033[1m3.\033[0m Considerar algoritmos mais robustos, como o K-Med

print('\n\033[1mFontes:\033[0m')
print('* Sensibilidade do K-Médias: https://scikit-learn.org/stable/modul
print('* Comparação K-Médias e K-Medoids: https://www.geeksforgeeks.org/k
print('* Lidando com Outliers: https://www.datageeks.com.br/outliers/')
```

Parte 4 - Exercício 3 - O algoritmo de K-médias é sensível a outliers nos dados. Explique.

Por que o K-Means é sensível a outliers?
* O algoritmo calcula os centróides como a média dos pontos no cluster.
* Outliers, que são valores muito distantes, podem deslocar o centróide, t
ornando-o menos representativo para a maioria dos pontos.

Impacto nos agrupamentos:
* Os clusters podem ficar distorcidos, não refletindo bem os dados.
* Em alguns casos, os outliers podem até formar clusters próprios, atrapal
hando a análise.

Como evitar o problema?
1. Remover ou tratar os outliers antes de rodar o algoritmo.
2. Normalizar os dados para reduzir o impacto de valores extremos.
3. Considerar algoritmos mais robustos, como o K-Medoids, que usa pontos r
eais como representantes dos clusters.

Fontes:
* Sensibilidade do K-Médias: https://scikit-learn.org/stable/modules/clust
ering.html#k-means
* Comparação K-Médias e K-Medoids: https://www.geeksforgeeks.org/k-means-v
s-k-medoids-clustering/
* Lidando com Outliers: https://www.datageeks.com.br/outliers/

In [26]:
```
# Parte 4 - Exercício 4 - Por que o algoritmo de DBScan é mais robusto à

print('\033[1mParte 4 - Exercício 4 - Por que o algoritmo de DBScan é mai

print('\033[1mPor que o DBSCAN é mais robusto a outliers?\033[0m')
print('* O algoritmo identifica automaticamente os outliers como pontos q
print('* Ele forma clusters com base na densidade de pontos, ignorando os
print('* Diferente do K-Means, o DBSCAN não usa centróides, então outlier
print('* Pontos isolados são classificados como ruído e não afetam o agru

print('\n\033[1mFontes:\033[0m')
print('* Robustez do DBSCAN: https://scikit-learn.org/stable/modules/clus
print('* Comparação entre DBSCAN e K-Means: https://towardsdatascience.co
print('* Lidando com Outliers: https://www.datageeks.com.br/outliers/')
```

Parte 4 – Exercício 4 – Por que o algoritmo de DBScan é mais robusto à presença de outliers?

Por que o DBSCAN é mais robusto a outliers?
* O algoritmo identifica automaticamente os outliers como pontos que não pertencem a nenhum cluster.
* Ele forma clusters com base na densidade de pontos, ignorando os que estão isolados.
* Diferente do K-Means, o DBSCAN não usa centróides, então outliers não deslocam os clusters.
* Pontos isolados são classificados como ruído e não afetam o agrupamento.

Fontes:
* Robustez do DBSCAN: https://scikit-learn.org/stable/modules/clustering.html#dbscan
* Comparação entre DBSCAN e K-Means: https://towardsdatascience.com/dbscan-clustering-explained-97556a2ad556
* Lidando com Outliers: https://www.datageeks.com.br/outliers/